EXHIBIT 16

Red Hat

Red Hat Blog Menu

The History of Containers

August 28, 2015 | Tim Hildred

SHARE

< Back to all posts

Given the recent massive spike in interest in Linux Containers, you could be forgiven for wondering, "Why now?". It has been argued that the increasingly prevalent cloud computing model more closely resembles hosting providers than traditional enterprise IT, and that containers are a perfect match for this model.

Red Hat named a Leader in the 2023 Gartner® Magic Quadrant™

Red Hat was positioned highest for ability to execute and furthest for completeness of vision in the Gartner 2023 Magic Quadrant for Container Management.

Read the report

Despite the sudden ubiquity of container technology, like so much in the world of open source software, containerization depends on a long series of previous innovations, especially in the operating system. “One cannot resist an idea whose time has come.” Containers are such an idea, one that has been a long time coming.

Early Days

The year 2000 was a busy one in the world of computing. 15 years ago, Bill Gates stood aside for Steve Balmer at Microsoft. The NASDAQ Composite stock market index peaked at 5132.52, the beginning of the end for the dot-com boom. The patent on the RSA cryptographic algorithm ended and the last Multics (Multiplexed Information and Computing Service) got turned off. And jails, an early implementation of container technology, was added to FreeBSD.

In 2001, container technology made it to the Linux side of the house. Jacques Gélinas created the VServer project, which according to the 0.0 version’s change log allowed “running several general purpose Linux server on a single box with a high degree of Independence and security.”

The Linux-VServer solution was the first effort on Linux to “separate the user-space environment into distinct units (Virtual Private Servers) in such a way that each VPS looks and feels like a real server to the processes contained within.” And though the Linux-VServer project lacked process migration and clustering, its real weakness was that it required a patched kernel, imposing an additional overhead on distributors and system administrators (or is it a feature?).

Incremental Innovations

It was Paul Menage’s approach in 2006 of adapting the cpusets mechanism already in the mainline kernel that really moved containerization on Linux forward, requiring minimally intrusive changes with little impact on performance, code quality, complexity, and future compatibility.

The result was generic process containers, which were later renamed control groups, or cgroups, to reflect the fact that “this code is an important part of a container solution... it's far from the whole thing.” Cgroups allow processes to be grouped together, and ensure that each group gets a share of memory, CPU and disk I/O; preventing any one container from monopolizing any of these resources.

Under former Red Hat Linux kernel developer and a principal software engineer Tejun Heo’s stewardship, cgroups underwent a massive redesign, replacing multiple, per-controller cgroup hierarchies with a “single kernel cgroup hierarchy... [that] allow[s] controllers to be individually enabled for each cgroup” and is the “private property of systemd.” These changes, especially when combined with functionality in systemd, increased the consistency and manageability of cgroups.

Kernel namespaces are another key part of a container solution, with Red Hatter Eric W. Biederman’s 2008 user namespaces patches being arguably the most complex and one of the most important namespaces in the context of containers. The implementation of user namespaces allows a process to have it’s own set of users and in particular to allows a process root privileges inside a container, but not outside.

Security Concerns

The Linux Containers project (LXC), created by engineers from IBM around 2008, layered some userspace tooling on top of cgroups and namespaces. While the LXC provided an improved user experience around containers, it had some people asking “Are LXC containers enough?”

Of particular concern was security, because the “DAC (discretionary access control) system on which LXC [originally] relie[d] for all security is known to be incomplete and so it is entirely possible to accidentally/intentionally break out of the container and/or impose a DOS attack on the host OS.”

Since the 1.0 release of LXC in early 2014, the situation improved as LXC began leveraging some longstanding Linux security technologies. In addition to security features in cgroups and namespaces, support for SELinux and Seccomp were added.

Seccomp is a Linux kernel feature by Red Hat’s Andrea Arcangeli for limiting the system calls which a task can use. The intent was to allow underutilised CPU to be rented out to untrusted guests without fearing they’d abuse other resources, an idea that maps to the container use case.

SELinux is the mandatory access control (MAC) system developed by a consortium of companies and government agencies that is really good at “labeling processes, files, and devices [and] at defining rules on how labeled processes interact with labeled processes, files, and devices.” Linux containers, as a group of processes, are a good match for hardening with SELinux.

Common Packaging Format

The genesis of the current Linux Container craze is of course the open source Docker project and associated Docker container format. Docker built on all of the incremental developments that came before it and upped the ante by (originally) wrapping the LXC userspace tools with even easier to use tooling aimed at developers looking for simple ways to package their applications.

In June 2015, Docker the company, the largest contributor to Docker the project (Red Hat is the second), donated the project's existing codebase to the Open Container Initiative, a lightweight governance structure under the auspices of the Linux Foundation created to prevent fragmentation and promote open standards by "cloud giants" including Red Hat. Maintainers of both libcontainer and appc, the donated codebase, form a technical advisory board that will guide and drive the project. Red Hat is proud to have one of our own, Vincent Batts, helping to guide the Initiative's technical direction.

Orchestrating at Scale

One of the biggest challenges for those working with containers today is how to deploy and orchestrate containers at scale. While a number of approaches are emerging, the one that is gaining the most traction is Kubernetes. When Kubernetes launched in 2014, Google discussed how "everything at Google runs in a container" to support their various service offerings and made news when they revealed that they were starting "over 2 billion containers per week." Google outlined details on the lineage of the Kubernetes project which is traced from Borg, Google's internal container cluster-management system.

Over the past year, Red Hat has contributed substantially to Kubernetes in various areas, as we work to bring the concepts of atomic, immutable infrastructure to enterprise customers in products like OpenShift, Red Hat Enterprise Linux and RHEL Atomic Host. Many of the Google developers that Red Hat collaborates with today in the Kubernetes community were previously developers on the Borg project. As we work with Google and others to bring Kubernetes to enterprise users, we benefit greatly from their experience as they bring the best ideas from Borg into Kubernetes and learn from its shortcomings.

Looking Ahead

Despite the long history of incremental innovations, recent advancements in Linux around Linux containers are revolutionizing the way that companies will develop, consume, and manage applications. As with traditional applications, containerized applications interact with and depend on the operating system.

Future innovations in the containerization of applications will build on incremental improvements to the Linux operating system, as they have all along. Red Hat's container strategy helps IT deliver applications to accelerate business agility, developer productivity, and deployment flexibility across hybrid cloud environments.

ABOUT THE AUTHOR

Tim Hildred

Principal Communications Strategist

Tim Hildred is a Principal PnT Communications Strategist at Red Hat.

Read full bio →

More like this

BLOG POST

Friday Five — July 5, 2024 | Red Hat

BLOG POST

Sharing is caring: How to make the most of your GPUs (part 1 - time-slicing)

ORIGINAL SHOWS

The Truth About Netcode | Compiler

ORIGINAL SHOWS

Transforming Your Acquisition

Browse by channel

Explore all channels →

Automation

The latest on IT automation for tech, teams, and environments

Artificial intelligence

Updates on the platforms that free customers to run AI workloads anywhere

Open hybrid cloud

Explore how we build a more flexible future with hybrid cloud

Security

The latest on how we reduce risks across environments and technologies

Edge computing

Updates on the platforms that simplify operations at the edge

Infrastructure

The latest on the world's leading enterprise Linux platform

Applications

Inside our solutions to the toughest application challenges

Original shows

Entertaining stories from the makers and leaders in enterprise tech

Red Hat

Products

Tools

Try, buy, & sell

Communicate

About Red Hat

We're the world's leading provider of enterprise open source solutions—including Linux, cloud, container, and Kubernetes. We deliver hardened solutions that make it easier for enterprises to work across platforms and environments, from the core datacenter to the network edge.

Select a language

English

About Red Hat

Jobs

Events

Locations

Contact Red Hat

Red Hat Blog

Diversity, equity, and inclusion

Cool Stuff Store

Red Hat Summit

Privacy statement

Terms of use

All policies and guidelines

Digital accessibility

Cookie Preferences and Do Not Sell or Share My Personal Information